Universidad Autónoma de Baja California

Facultad de Ingeniería, Arquitectura y Diseño

Practica 3

Análisis de sentimientos

**Mauricio Santiago Valdovinos Morales**

**Carlos Xavier Gallardo Rosas**

**Sergio Alberto Arce Ruelas**

Materia: Inteligencia Artificial

Profesor: Hugo Armando Guillen Ramírez

Ensenada, Baja California a; 31 de Mayo del 2018.

Introducción

En la actualidad la inteligencia artificial es uno de los temas que más atención está captando en el mundo tecnológico y en el de los negocios. Lo que nos inspiró para elegir para elegir el tema de este proyecto fue la película de “Imitation game” donde sale uno de los padres de la inteligencia artificial Alan Turing y se menciona algo llamado el *“Test de Turing”*. La idea inicial para el proyecto era realizar ese test pero era un trabajo muy complejo y ambicioso de realizar), por lo que se decidió realizar una parte de lo que conlleva todo el trabajo de cumplir con ese examen. Realizamos un análisis de sentimientos utilizando técnicas de machine learning y un poco de procesamiento de lenguaje natural (PLN).

Procesamiento de lenguaje natural

El **procesamiento del lenguaje natural** (PLN) es el campo que combina las tecnologías de la ciencia computacional (como la inteligencia artificial, el aprendizaje automático o la inferencia estadística) con la lingüística aplicada, con el objetivo de hacer posible la comprensión y el procesamiento asistidos por ordenador de información expresada en lenguaje humano para determinadas tareas, como la traducción automática, los sistemas de diálogo interactivos, el análisis de opiniones, etc. [1]

Objetivo

El objetivo principal de la elaboración de este proyecto era determinar si una película era buena o no según la información de las críticas de las películas mediante un programa inteligente que utilizaba los métodos de PLN para poder saber si el contenido review de la película era una crítica positiva o negativa y según la tendencia de estas críticas se determina la calificación de la película.

¿Cómo procesa los textos?

Para poder procesar la información obtenida y determinar los sentimientos de estos utilizamos una librería muy poderosa para el PLN llamada NLTK. Esta paquetería cuenta con diferentes funciones para poder realizar todo el procesado de información. El procesado que se le dio a la información fue con los siguientes pasos.

1.- Tokenizing

Para poder analizar un texto este debe poder ser dividido entre sus oraciones o incluso entre palabras, para poder así analizar toda la estructura del texto. Con el metodo de tokenizing esto es posible, dividiendo un párrafo en sus oraciones o en sus palabras.

2.- Stop words

Dentro de todas las oraciones o párrafos existen diversas palabras, pero existen unas palabras que para una computadora no tienen importancia como lo son aquel, a, aun, atrás, etc. Estas palabras son las llamadas stop words, conociendo estas palabras podemos filtrar nuestra información de entrada para quedarnos solo con las palabras que nos importan para después realizar un análisis con estas.

Clasificadores

Naive Bayes

Es una técnica de clasificación basada en  el teorema de Bayes con un supuesto de independencia entre los predictores. En términos simples, un clasificador de Naive Bayes supone que la presencia de una característica particular en una clase no está relacionada con la presencia de ninguna otra característica. Por ejemplo, una fruta puede considerarse una manzana si es roja, redonda y de aproximadamente 3 pulgadas de diámetro. Incluso si estas características dependen unas de otras o de la existencia de otras características, todas estas propiedades contribuyen de forma independiente a la probabilidad de que esta fruta sea una manzana y es por eso que se la conoce como "ingenua".

El modelo Naive Bayes es fácil de construir y particularmente útil para conjuntos de datos muy grandes. Junto con la simplicidad, se sabe que Naive Bayes supera incluso a los métodos de clasificación altamente sofisticados.[3]

Multinomial Naive Bayes

El clasificador multinomial Naive Bayes es adecuado para la clasificación con características discretas (por ejemplo, recuentos de palabras para la clasificación de texto). La distribución multinomial normalmente requiere recuentos de características enteros. [4]

Clasificador Bernoulli Naive Bayes

En el modelo de eventos Bernoulli multivariante, las características son booleanos independientes (variables binarias) que describen las entradas. Al igual que el modelo multinomial, este modelo es popular para tareas de clasificación de documentos, donde la ocurrencia de términos binarios (es decir, una palabra se produce en un documento o no) se utilizan en lugar de frecuencias de términos (es decir, la frecuencia de una palabra en el documento). [5]

SGDC

Este estimador implementa modelos lineales regularizados con aprendizaje de pendiente descendente estocástica (SGD): el gradiente de la pérdida se estima cada muestra a la vez y el modelo se actualiza a lo largo del camino con un programa de fuerza decreciente (también conocido como tasa de aprendizaje). SGD permite el aprendizaje de minibatch (en línea / fuera del núcleo).

Esta implementación funciona con datos representados como matrices densas o dispersas de valores de coma flotante para las características. El modelo que ajusta se puede controlar con el parámetro de pérdida; de forma predeterminada, se adapta a una máquina de vector de soporte lineal (SVM). [6]

CSVC

C-Support Vector Classification.

La implementación se basa en libsvm. La complejidad del tiempo de ajuste es más que cuadrática con el número de muestras, lo que hace que sea difícil escalar al conjunto de datos con más de un par de 10000 muestras.

El soporte de multiclass se maneja según un esquema de uno contra uno.

Para obtener detalles sobre la formulación matemática precisa de las funciones del kernel proporcionadas y cómo se afectan entre sí gamma , coef0 y grado , consulte la sección correspondiente en la documentación narrativa:[Funciones del kernel](http://scikit-learn.org/stable/modules/svm.html#svm-kernels) . [7]

LINEAR SVM

Un modelo SVM es una representación de los ejemplos como puntos en el espacio, mapeados de manera que los ejemplos de las categorías separadas estén divididos por un espacio libre lo más ancho posible. A continuación, se mapean nuevos ejemplos en ese mismo espacio y se predice que pertenecen a una categoría según el lado del espacio en el que caen.

Además de realizar la clasificación lineal, las SVM pueden realizar de manera eficiente una clasificación no lineal utilizando lo que se conoce como el truco del kernel , mapeando implícitamente sus entradas en espacios de características de alta dimensión. [8]

NUSVC

Optimiza un clasificador de vector de soporte para el conjunto de datos A mediante programación cuadrática. La diferencia con la rutina estándar de SVC es el uso y la interpretación del parámetro de regularización NU. Es un límite superior para el error de clasificación esperado. Por defecto, NU se estima mediante el leave-one-error de la regla 1\_NN . Para NU = NaN, se realiza una optimización automática utilizando REGOPTC .

Si KERNEL = 0 se supone que A ya es la kernelmatrix (square). En este caso, también se debe suministrar una matriz B del kernel en la evaluación por B \* W o PRMAP (B, W) .

Hay varias formas de definir KERNEL , por ejemplo, PROXM ([] , 'r' , 1) para un kernel de base radial o por USERKERNEL para un kernel definido por el usuario.

SVC es básicamente un clasificador de dos clases. Los problemas de clase múltiple se resuelven en una forma de uno contra el resto por MCLASSC . Los clasificadores de base resultantes se combinan por la regla de confianza máxima. Un mejor combinador no lineal podría ser QDC , por ejemplo, W = A \* (SVC \* QDC ([], [] , 1e-6)).

[9]

REGRESION LOGISTICA

La regresión logística es otra técnica prestada por el aprendizaje automático del campo de la estadística.

Es el método de referencia para problemas de clasificación binarios (problemas con dos valores de clase).

La regresión logística usa una ecuación como representación, muy similar a la regresión lineal.

Los valores de entrada (x) se combinan linealmente utilizando pesos o valores de coeficientes (a los que se hace referencia como la letra mayúscula griega Beta) para predecir un valor de salida (y). Una diferencia clave con respecto a la regresión lineal es que el valor de salida modelado es un valor binario (0 o 1) en lugar de un valor numérico. [10]

Paqueteria necesaria para el programa

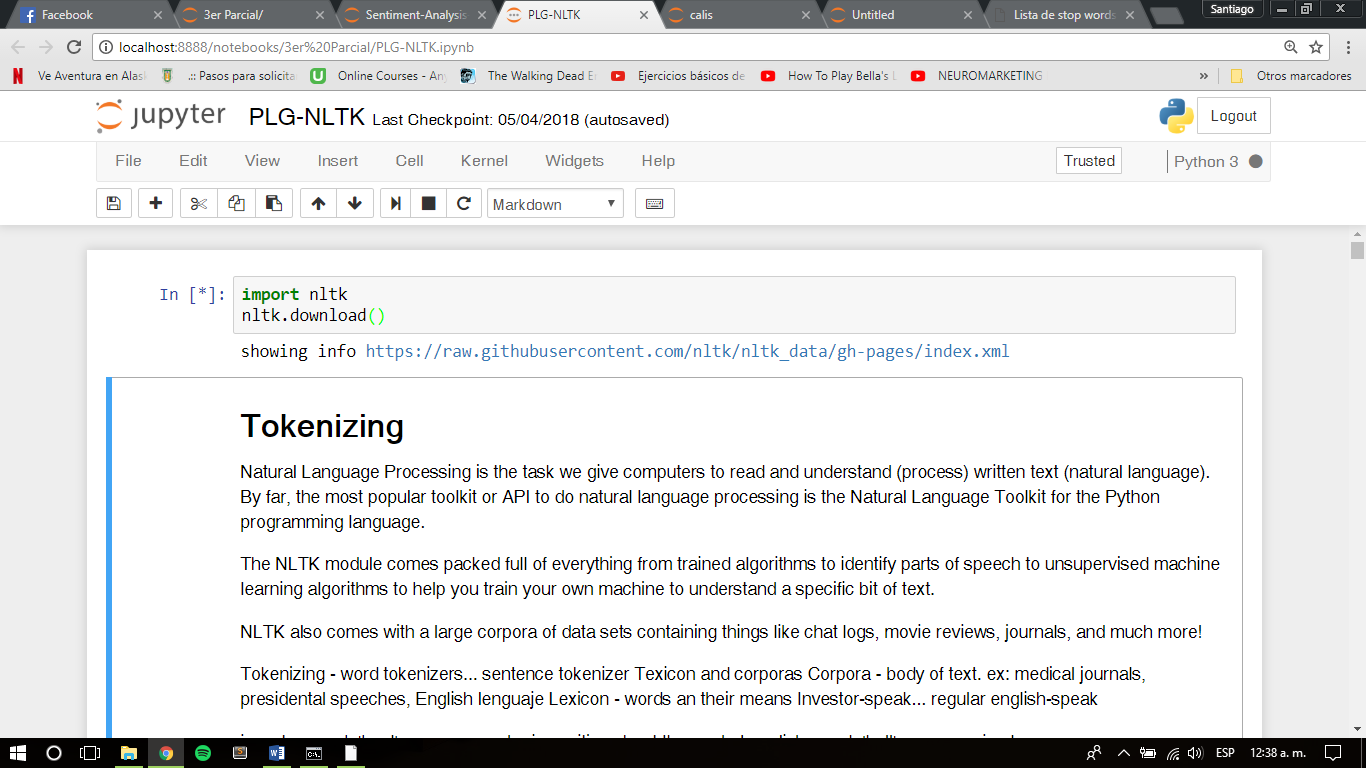
1.- Primero se debe realizar la instalación de diversas librerías como lo son NLTK, sklearn, matplotlib con la instrucción en la ventana de comandos

*pip install nltk*

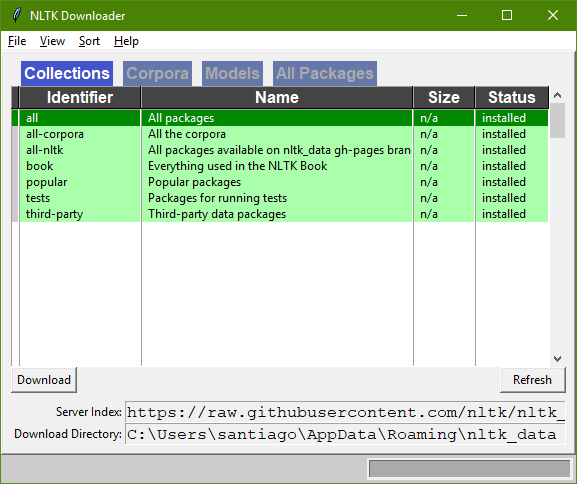
*pip install sklearn*

*pip install matplotlib*

Después de realizar la instalación se deben bajar las paqueterías de la librería NLTK. Esto puede realizarse ejecutando la siguiente instrucción:

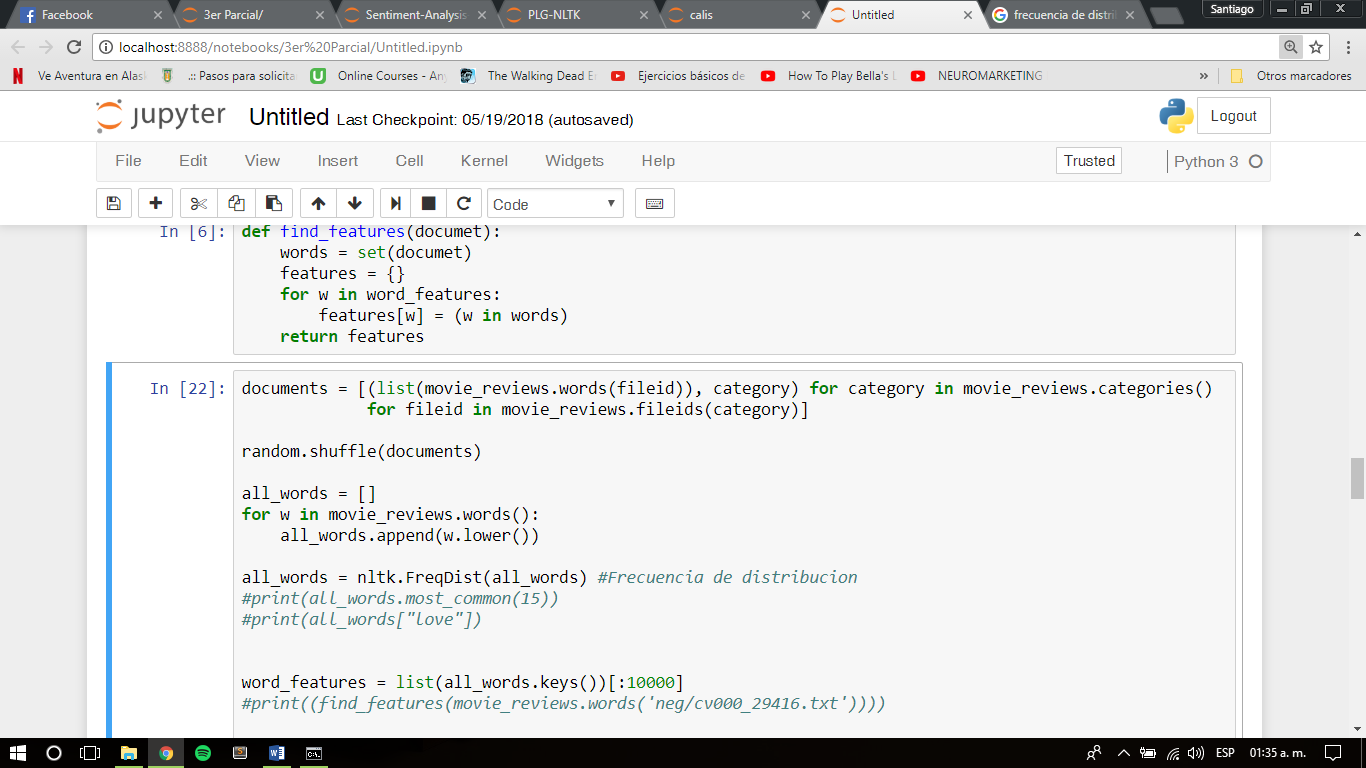


Una vez ejecutado se abrirá una ventana y se recomienda descargar e instalar todas las paqueterías faltantes, sobre todo la de corpora.

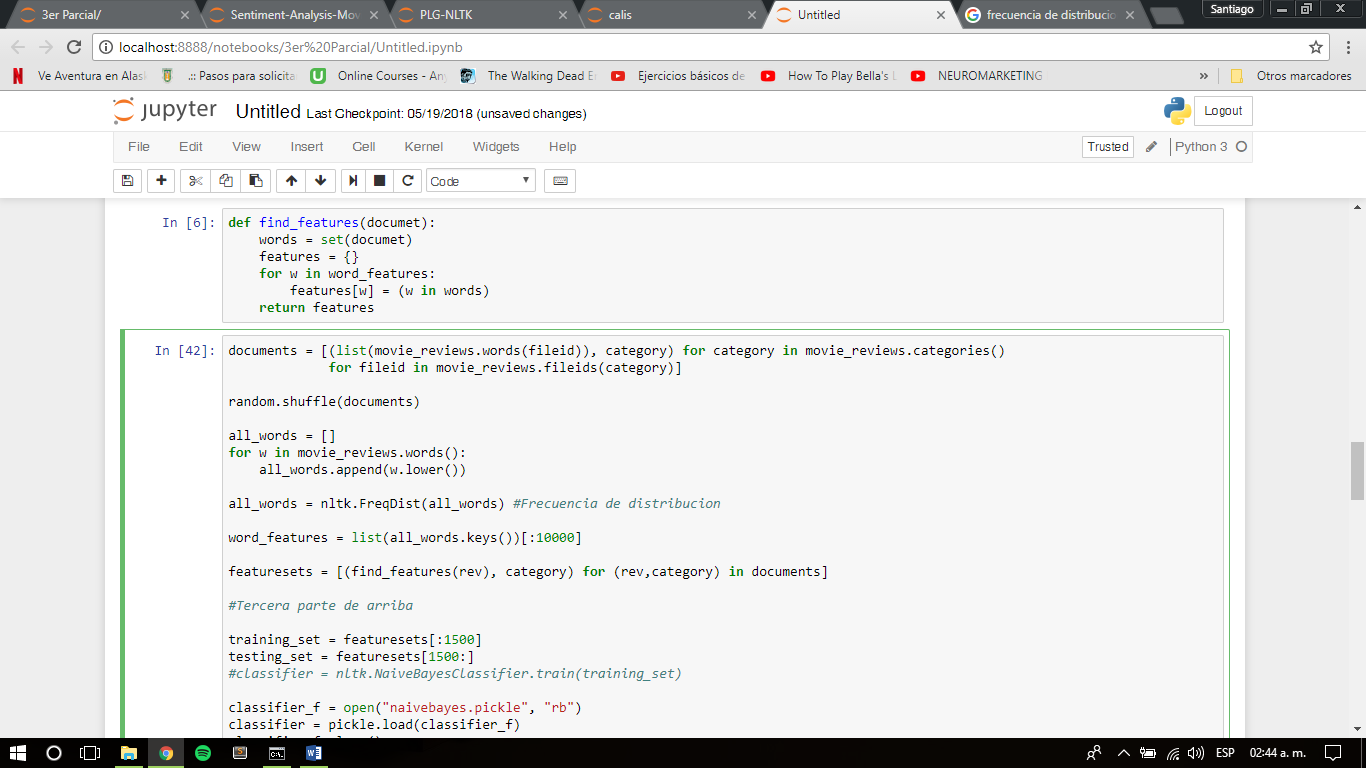


Procedimiento del programa

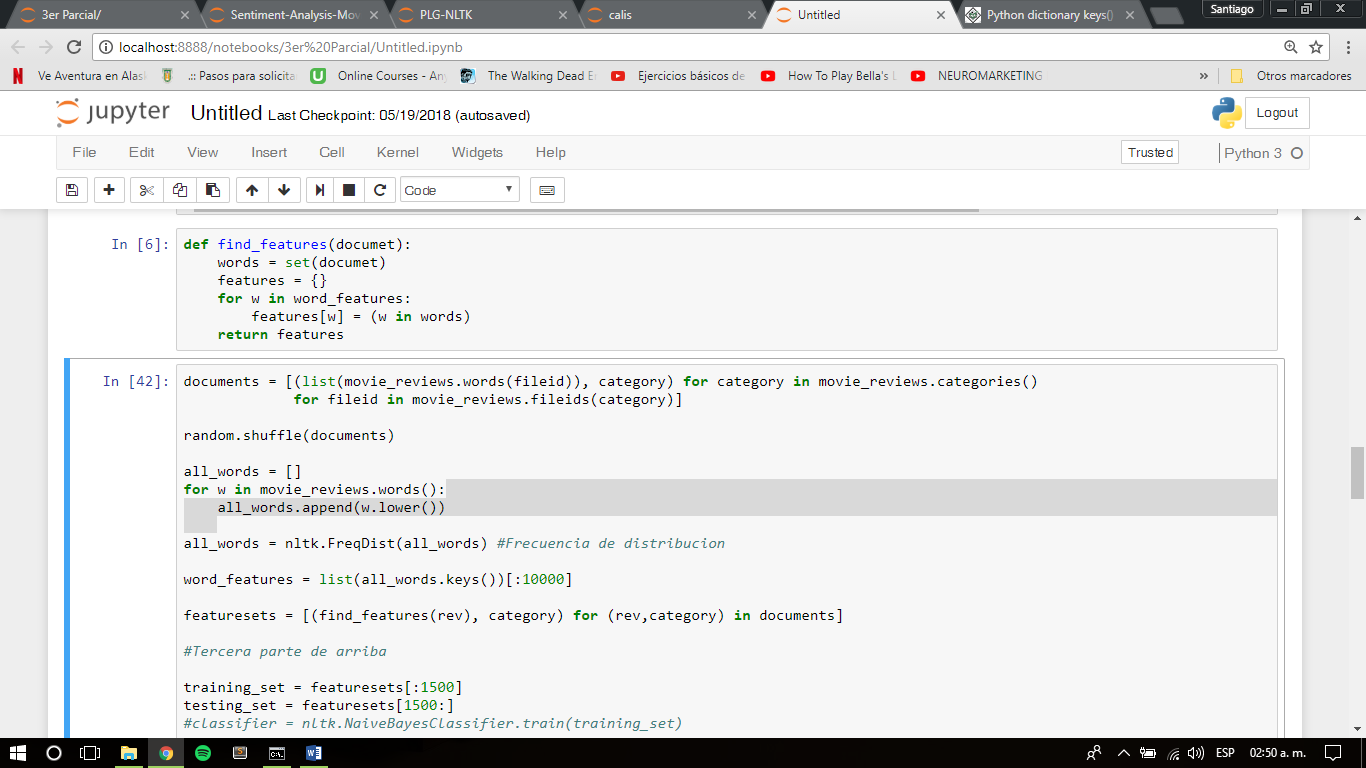
1.- El primer paso es obtener el data set, para esto en la paquetería de NLTK se cuenta con corpora y con esta colección se puede acceder a 2000 reviews, 1000 positivas y 1000 negativas para poder entrenar el algoritmo. Después de obtenerlas se revuelve la información para no darle al programa información continua de una misma categoría para tener un mejor entrenamiento.

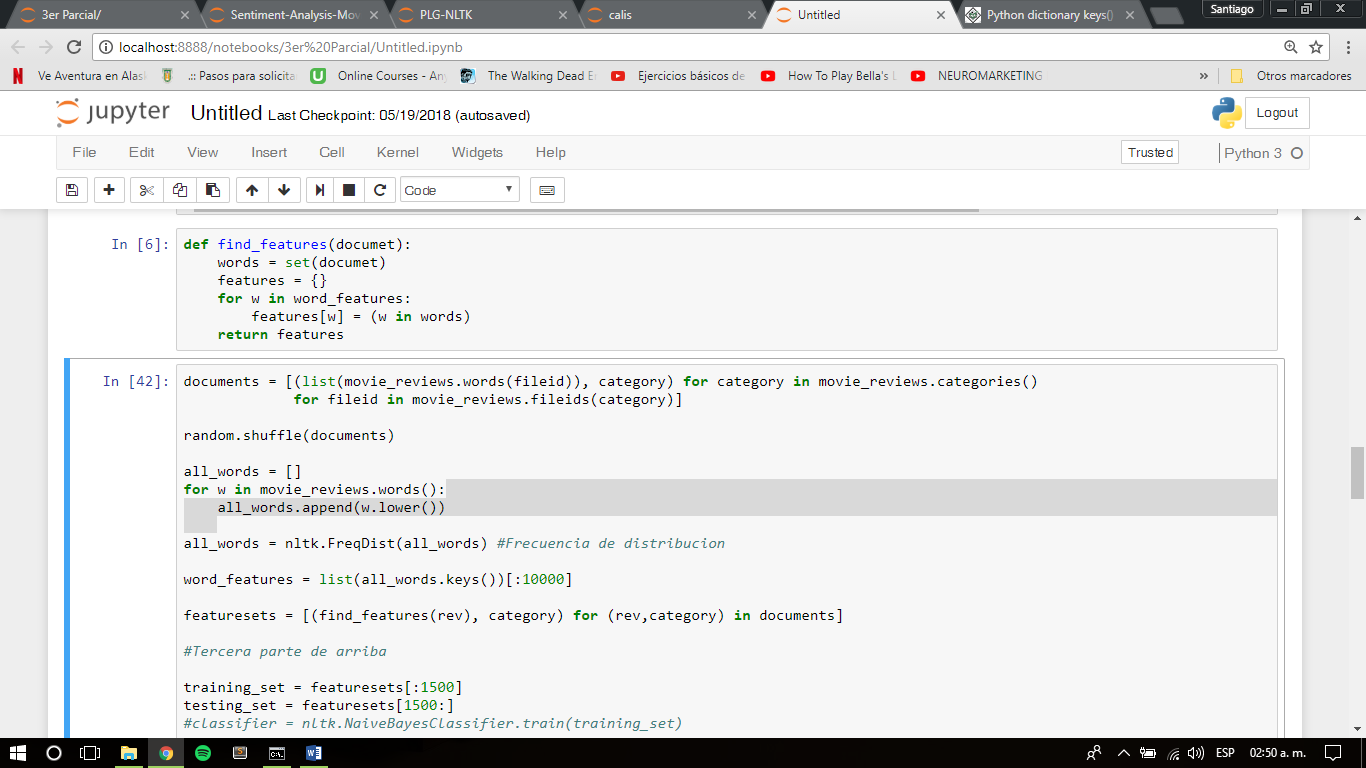


2.- Después se obtienen las palabras que hay en todas las reviews y utilizando una distribución de frecuencia se sacan todas las palabras repetidas.

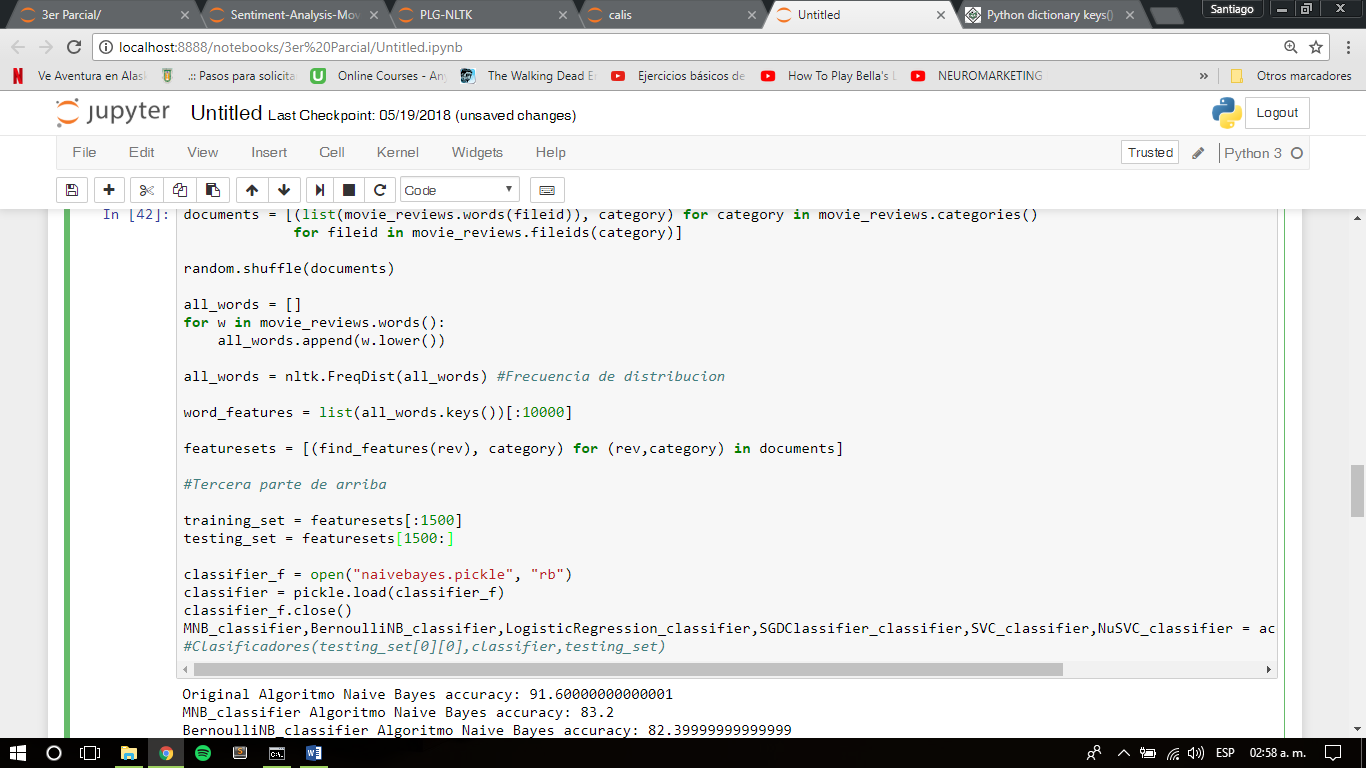


3.- Encontrar las características de las reviews, esto es utilizando las palabras de las reviews que el programa ya conoce.

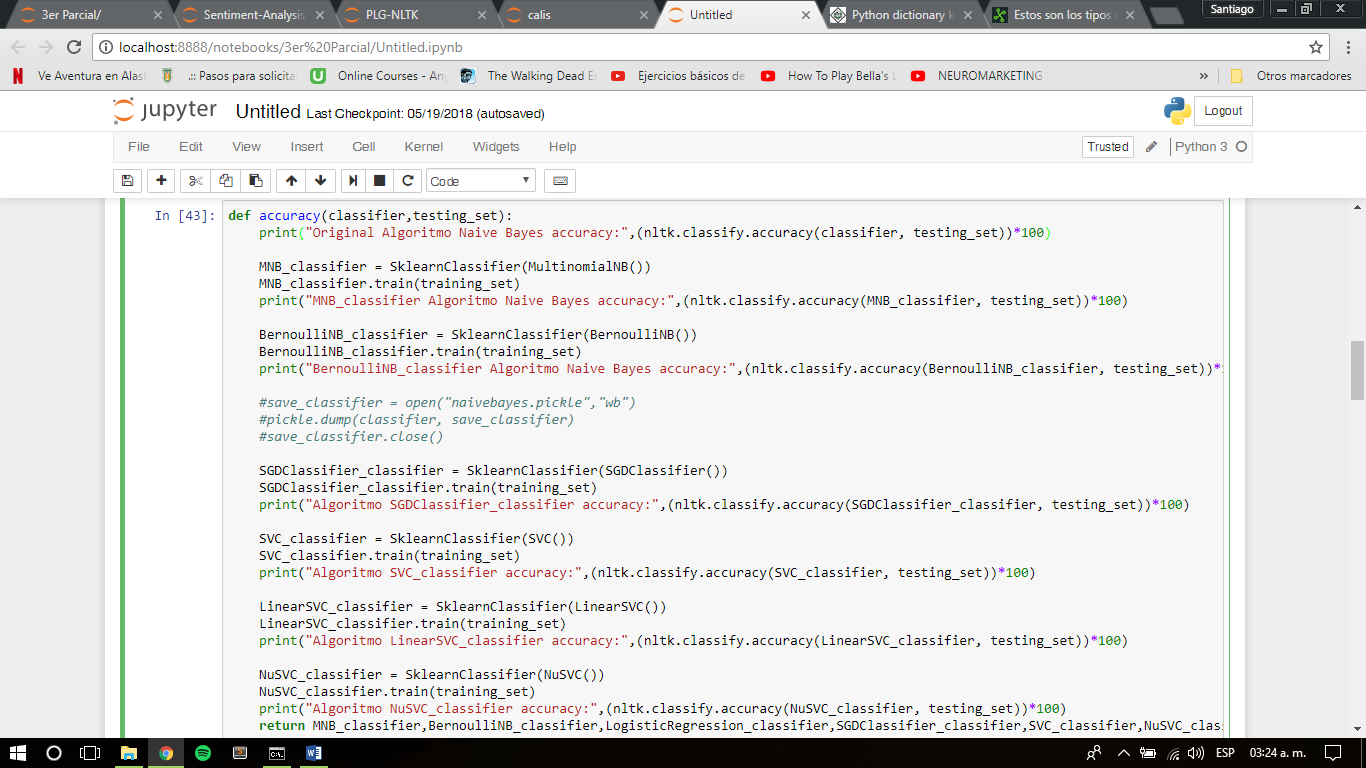




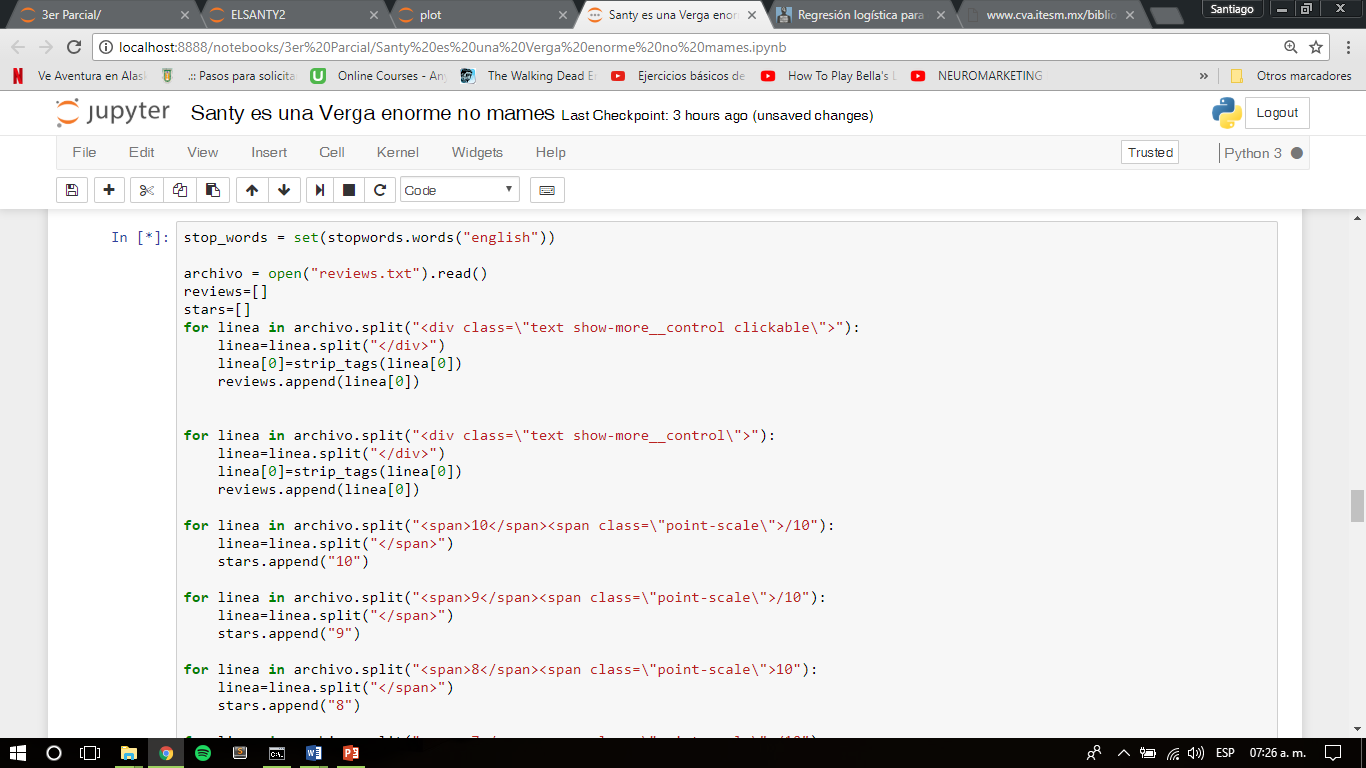
4.- Después de procesar toda la información de entrada es hora de hacer que el programa aprenda y para esto entrenamos al programa con 1500 reviews.

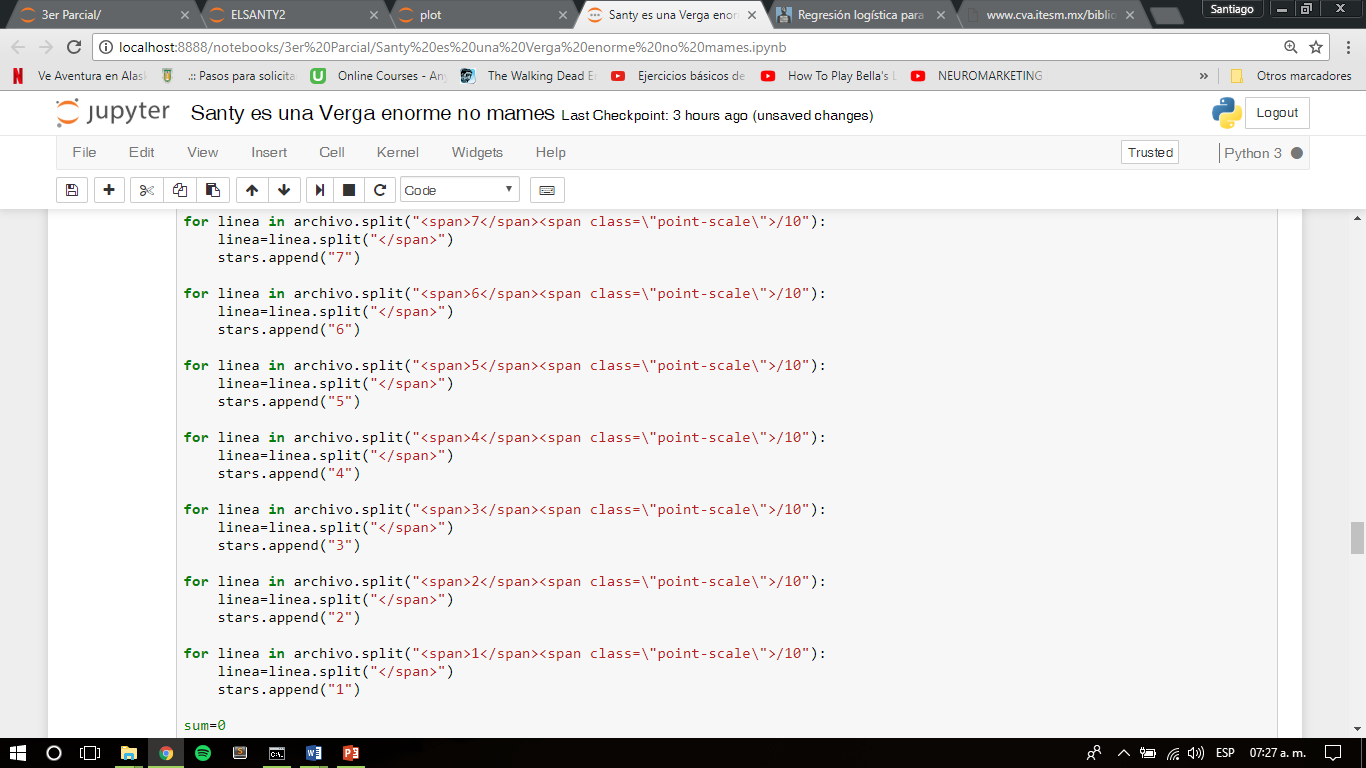


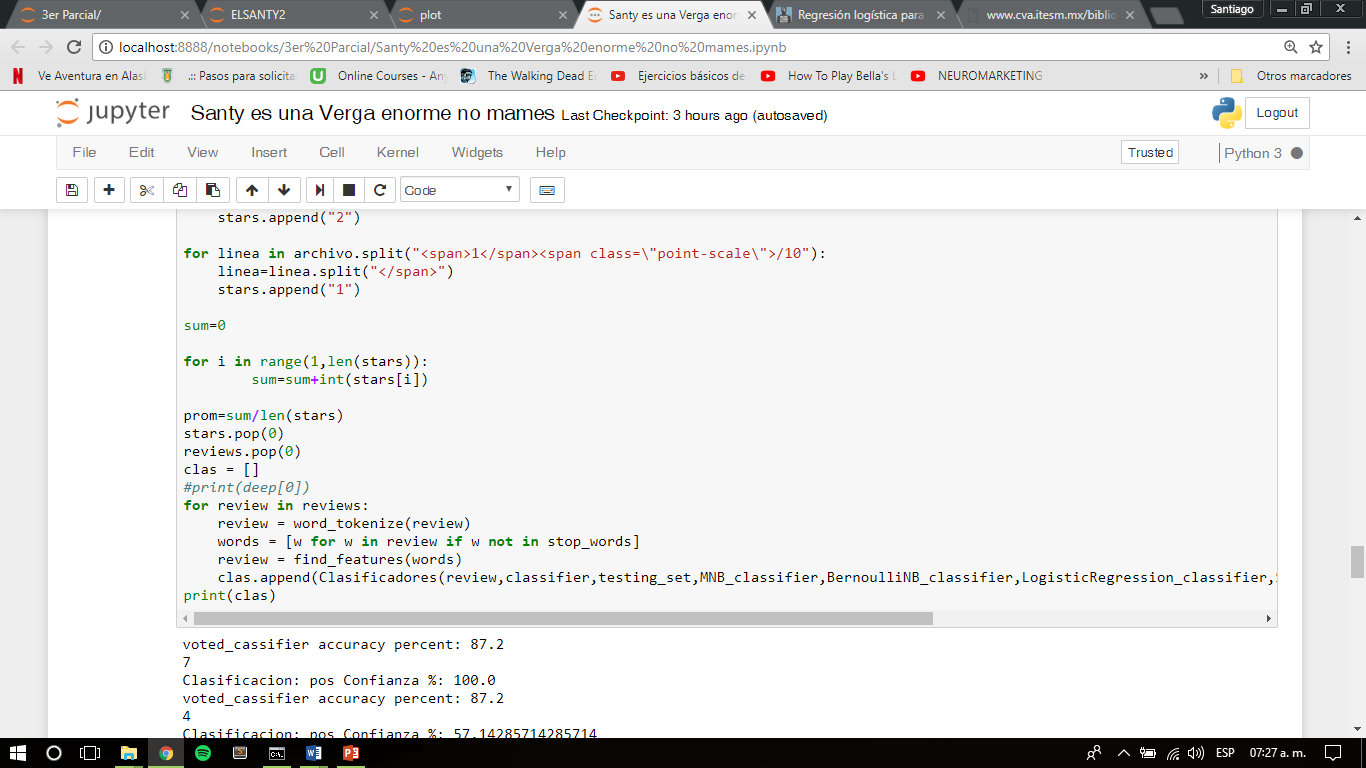
5.- Poner a prueba el algoritmo con 500 reviews de prueba, en nuestro caso utilizamos diversos tipos de clasificadores para obtener un mejor resultado.



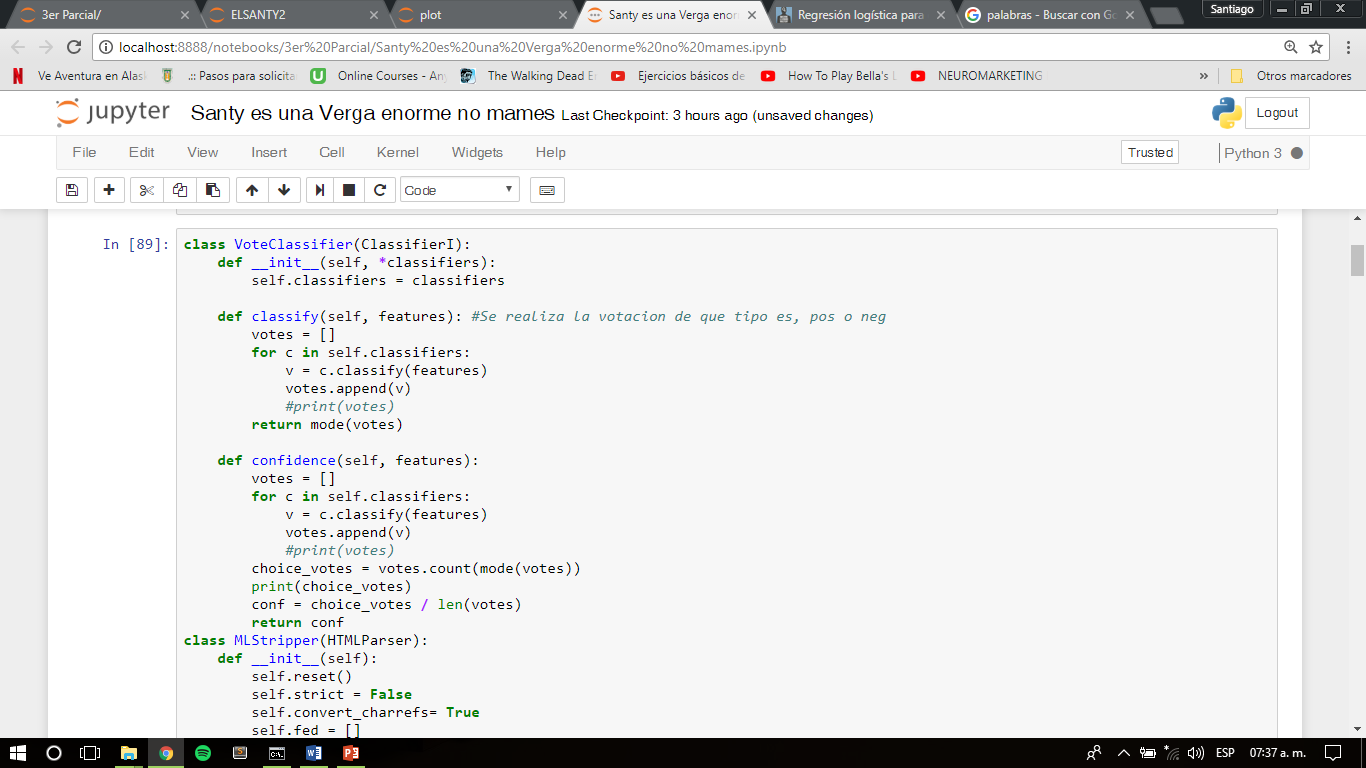
6.- Para realizar el objetivo de la práctica se realizó una búsqueda de reviews de la película “Infinity War” de la página de IMDb. Donde se extrajo el código html donde se encontraban las reviews y filtrando las etiquetas html extrajimos la informacion deseada.





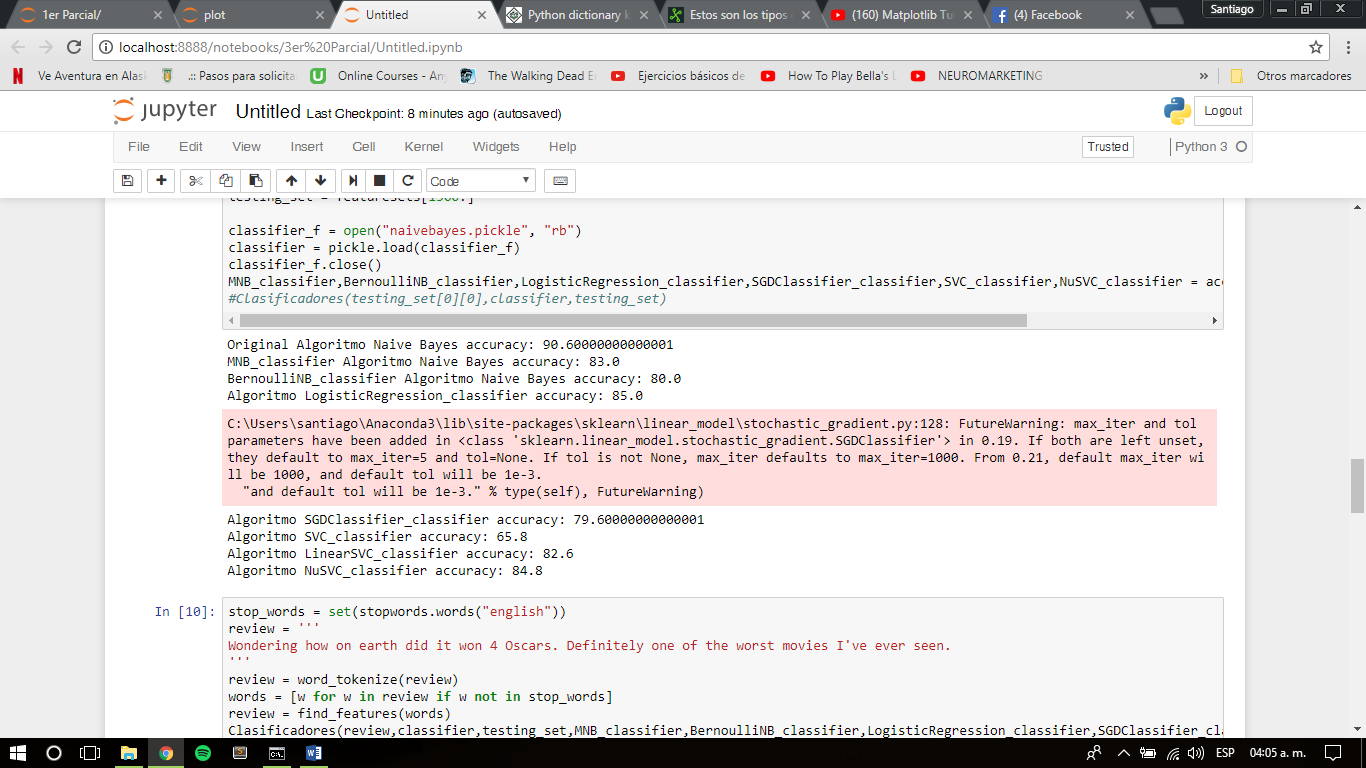


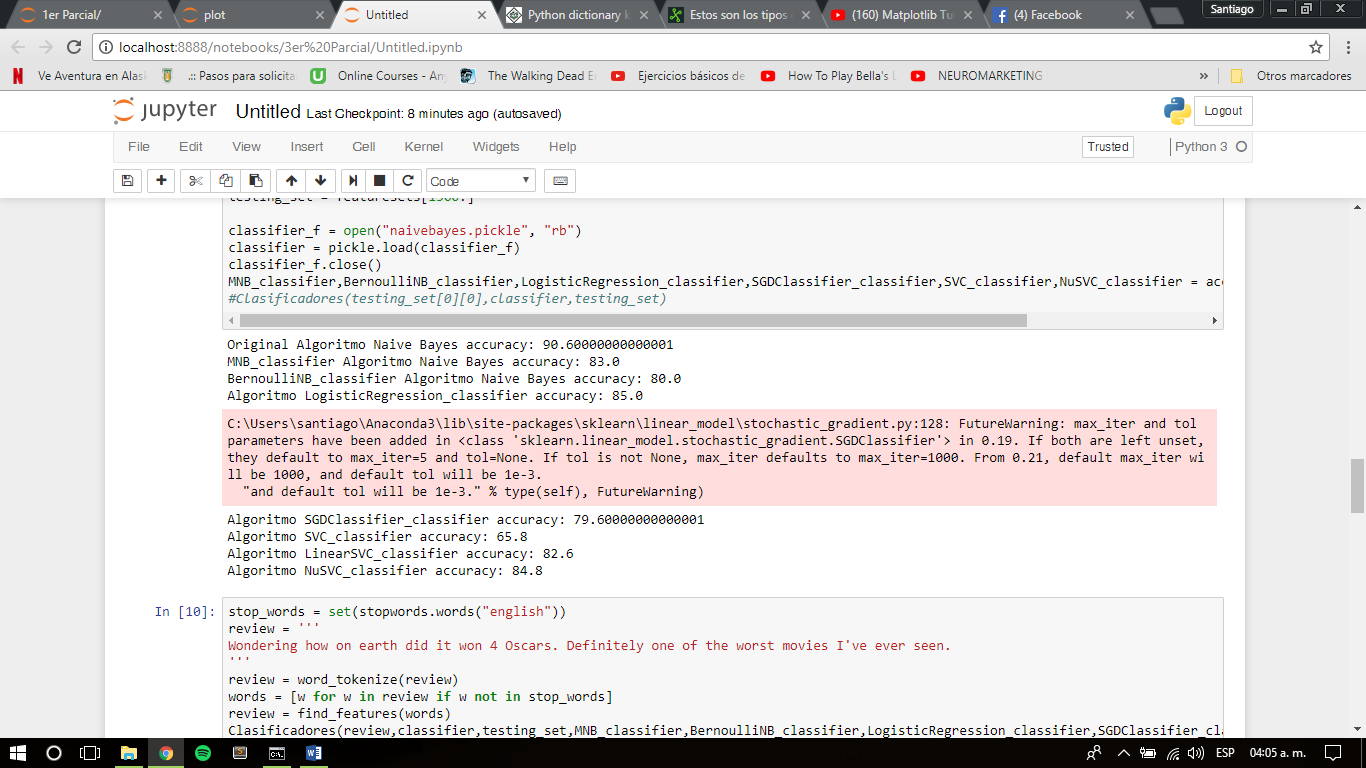
7.- Determinando si es buena o no la review. Esto se lleva a cabo por un metodo llamado ensamble el cual con todos los clasificadores que tenemos se toma una decisión votando por en que categoría clasifica la review y el resultado más votado será el valor elegido.

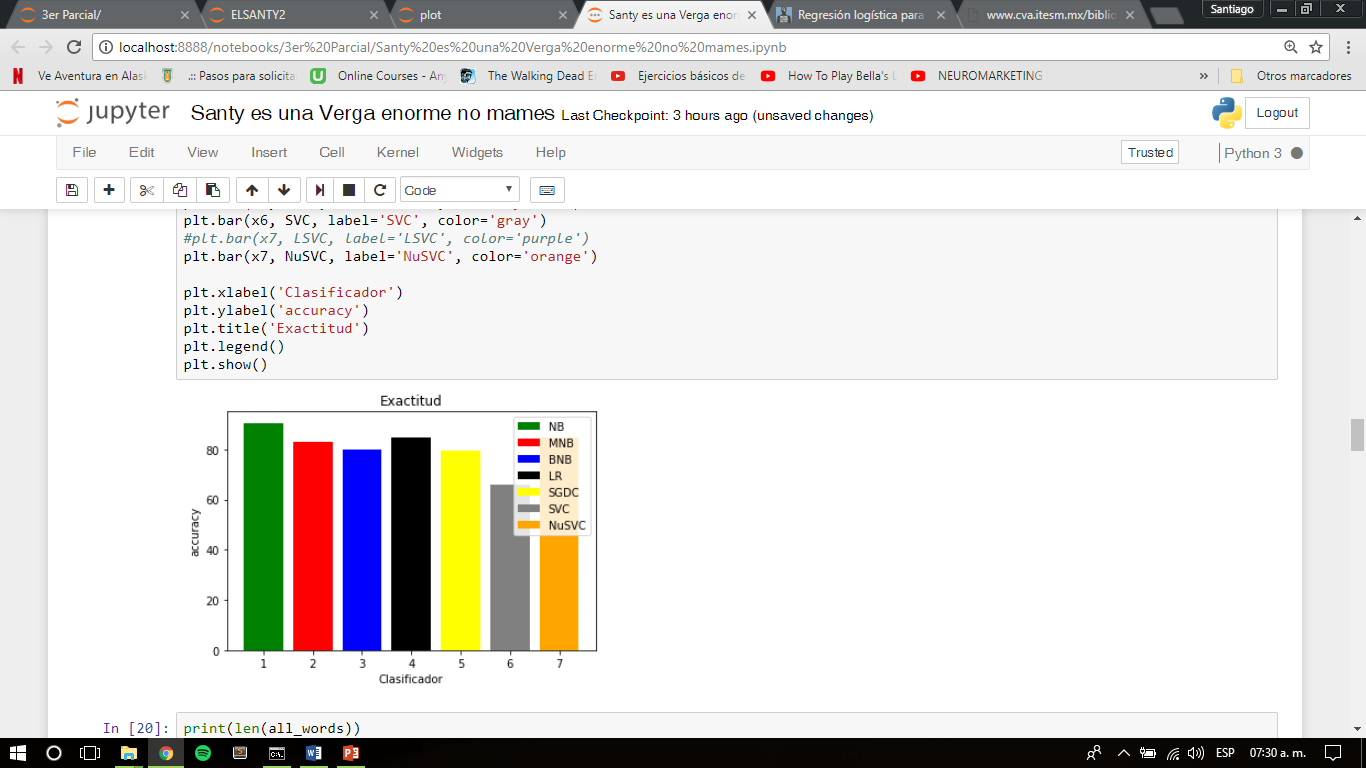


Resultados

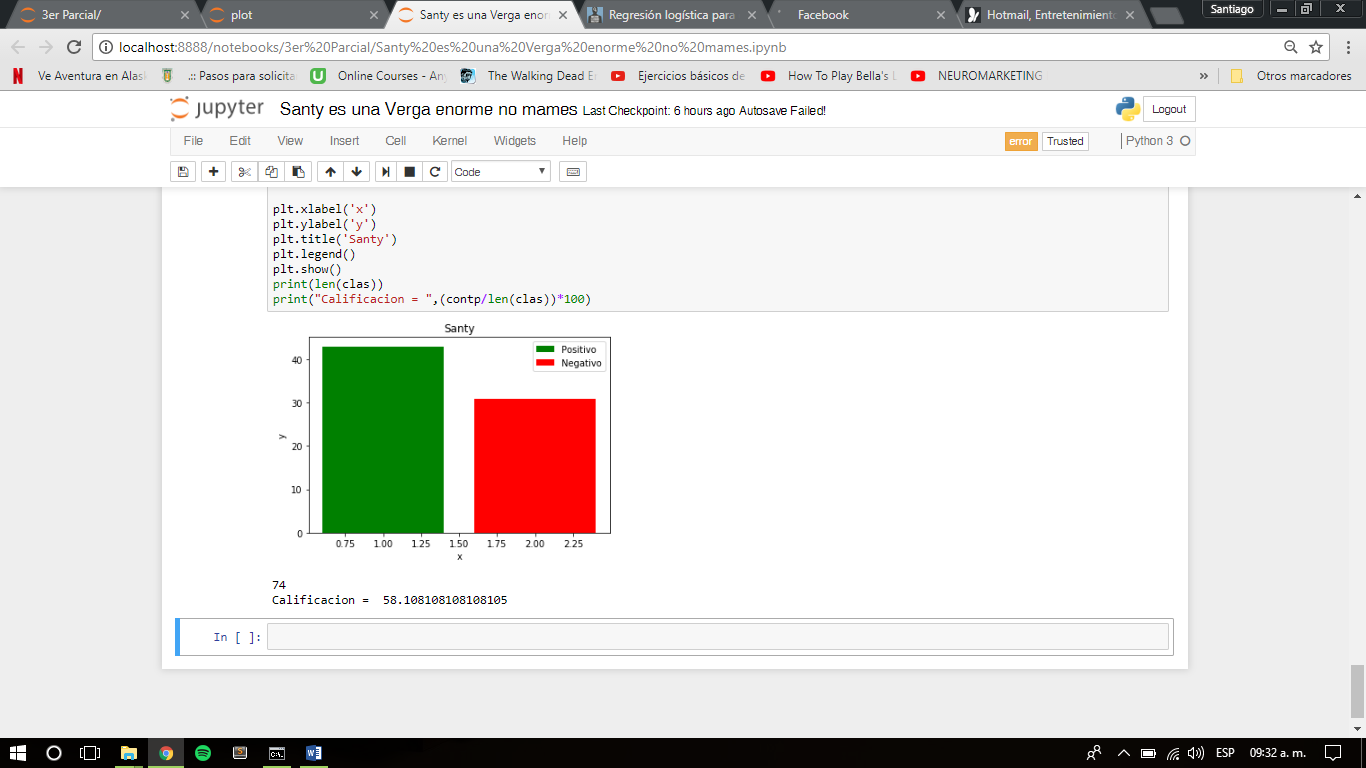
Valores de exactitud







Resultado final



Una calificación de la película de 58 según 74 reviews revisadas.

Problemas durante la práctica

1. La obtención de las reviews de Infinity war, fue el mayor problema ya que de donde se estaban adquiriendo era la página IMDb y esta página cuenta con mucha seguridad y era bastante complicado obtener las reviews de la película, pero esto se resolvió obteniendo el código html de la página que contiene las reviews y se filtró el código para llegar solo a el cometido de las reviews.
2. Lograr aumentar la exactitud del algoritmo ya que se obtenía un 72% pero utilizando el metodo de ensamble logramos aumentar hasta un 85% a 90% de exactitud

Conclusión

Referencias

1. Vicomtech. (2015). Procesamiento del Lenguaje Natural. 30/05/2018, de Vicomtech Sitio web: http://www.vicomtech.org/t4/e11/procesamiento-del-lenguaje-natural
2. Shantnu . (2016). Build a Sentiment Analysis app with Movie Reviews. 30/05/2018, de pythonforengineers Sitio web: http://pythonforengineers.com/build-a-sentiment-analysis-app-with-movie-reviews/
3. Anonimo. (2015). 6 sencillos pasos para aprender el algoritmo de Bayes ingenuo (con códigos en Python y R). 31/05/2018, de analyticsvidhya Sitio web: https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/
4. scikit-learn. (2017). MultinomialNB . 31/05/2018, de scikit-learn Sitio web: <http://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html>
5. Anonimo. (2017). Naive Bayes Classifiers. 31/05/2018, de Geeks for Geeks Sitio web: <https://www.geeksforgeeks.org/naive-bayes-classifiers/>
6. desarrolladores scikit-learn. (2007). Clasificadores lineales (SVM, regresión logística, ao) con entrenamiento SGD.. 2017, de desarrolladores scikit-learn Sitio web: <http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html>
7. desarrolladores scikit-learn. (2007). C-Support Vector Classification.. 2017, de desarrolladores scikit-learn Sitio web: <http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html>
8. scikit-learn developers. (2007). Support vector machines. 2017, de scikit-learn developers Sitio web: <http://scikit-learn.org/stable/modules/svm.html>
9. Anonimo. (2005). NUSVC. 2014, de Anonima Sitio web: http://www.37steps.com/prhtml/prtools/nusvc.html
10. Jason Brownlee. (2016). Regresión logística para el aprendizaje automático. 2018, de Machine Learning Mastery Sitio web: https://machinelearningmastery.com/logistic-regression-for-machine-learning/